

МИНОБРНАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ  
ВЛАДИВОСТОКСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И АНАЛИЗА ДАННЫХ  
КАФЕДРА ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И СИСТЕМЫ

# ОТЧЕТ ПО ПРАКТИКЕ НАД СОЗДАНИЕМ ТЕКСТОВОЙ RPG-ИГРЫ

Студент  
гр. БИН-25-1

\_\_\_\_\_

А.В.Наумова

Руководитель

\_\_\_\_\_

А.Д.Шнейдер

Владивосток 2026

## Введение

Данный проект предполагает разработку текстовой ролевой игры (RPG) для командной строки с использованием языка программирования Python. Целью работы является создание программной системы, реализующей ключевые механики жанра, включая управление персонажем, пошаговый бой, систему развития и инвентарь.

Проект структурирован как набор взаимосвязанных функциональных модулей:

1) создание и управления персонажем. Позволяет пользователю задать имя персонажа, получить случайно сгенерированные начальные характеристики (атака, защита, здоровье) и осуществить их ручное перераспределение

2) боевая система. Реализует пошаговый цикл, в котором игрок и противник совершают действия по очереди. Механика включает расчёт урона на основе значений атаки и защиты, а также выбор действия (стандартная атака, применение предмета, попытка побега). Противники обладают уникальными наборами характеристик

3) система прокачки. Обеспечивает начисление опыта за победу над противниками, повышение уровня персонажа и выдачу очков для улучшения его характеристик

4) инвентарь. Управляет коллекцией предметов (зелья, оружие, броня), которые предоставляют постоянные или временные модификаторы характеристик персонажа

5) система локаций. Организует простое нелинейное перемещение между узлами (локациями)

Таким образом, в рамках проекта будет создано консольное приложение, интегрирующее указанные модули в единую игровую систему. Реализация продемонстрирует применение основных парадигм и структур данных Python для построения логики пошаговой игры.



### 1.3 Инвентарь

Инвентарь – место, где Игрок может посмотреть на свои нынешние характеристики, уровень, количество опыта, свободные очки развития, надетую экипировку (всего слота три: Броня, Оружие и Аксессуар) и, конечно, имеющиеся предметы. Также, в инвентаре есть небольшая картинка протагониста для лучшего визуального эффекта.

Относительно функциональной его части, в меню инвентаря производится доступ поглощению и рассматриванию предметов, надеванию или, наоборот, снятию экипировки. При наличии свободных очков развития переход к их назначению также производится из инвентаря. Инвентарь представлен на рисунке 2.

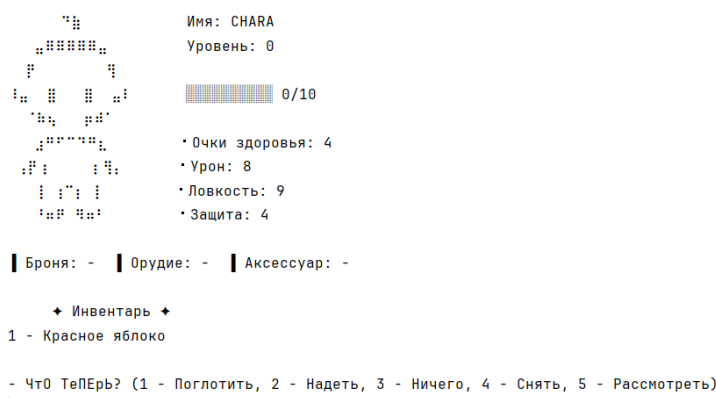


Рисунок 2 – Инвентарь

Инвентарь не ограничен по вместительности и при запуске игры содержит «Красное яблоко».

### 1.4 Локации и монстры

Всего в игре есть 5 локаций, каждая из которых имеет свой определённый набор встречаемых монстров и находимых предметов. Перемещение между ними происходит по желанию Игрока: между битвами у него есть возможность выбрать, остаться на месте или отправиться в другую, которую необходимо выбрать из двух случайных. Некоторые локации содержат более опасных монстров, чем другие, но это Игрок может выяснить только на собственном опыте.

Дополнительной локацией является «Поляна цветов» — безопасная зона, попадание в которую покажет картинку, сопровождаемую восстановлением 10 очков здоровья. Здесь невозможно встретить врагов или найти сокровище, поэтому после получения лечения с неё можно только уйти куда-то ещё.

То, как выглядит «Поляна цветов» в игре, представлено на рисунке 3.



Рисунок 3 – «Поляна цветов», вид в игре

Самих типов монстров в игре 14, у каждого есть свой набор характеристик и возможных к выпадению предметов. Особенность бабочки – враги с ней в группе атакуют первыми.

### 1.5 Бой

Бой ведётся пошагово. Если в группе монстров нет бабочки, первым действует Игрок. На своем ходу он может атаковать или попробовать сбежать. Процесс боя представлен на рисунке 4.

```

-
Гоблин-шаман УР 0
-----
Очки здоровья: 5
Урон: 10
Ловкость: 4
Защита: 4

- Что Делать Будем? (1 - Инвентарь, 2 - Атака, 3 - Побег, 4 - Отобразить монстров)
2
1 - Гоблин-шаман
- Кого Бьем?
1
- ВПеред!! Существо Гоблин-шаман нанесено 2 единиц Урона >:D

- -5 03 БАЙ Гоблин-шаман (・^・) осталось 1 03

```

Рисунок 4 – Процесс боя

После того как Игрок атаковал или провалил попытку побега, атакуют монстры, каждый нанося какое-то количество урона. Если защита Игрока больше атаки монстра, будет получена одна единица урона. Так же и в обратную сторону – врагу, имеющему большую защиту, чем урон у протагониста, будет нанесено от 1 до 2 единиц урона.

## 2 Разработка

### 1.1 Игрок

Игрок был реализован через класс `Player`. В нем содержатся все характеристики (в том числе инвентарь, экипированные предметы, нынешняя локация, уровень) и методы, относящиеся непосредственно к протагонисту (получить урон, поднять уровень и т. д.). Класс `Player` представлена на рисунке 5.

```
class Player:
    def __init__(self):...
    def get_stats(self):...
    def get_exp_bar(self):...
    def get_level(self):...
    def show_all(self):...
    def greet(self):...
    def re_stats(self):...
    def points(self):...
    def rest(self):...
    def get_hurt(self, enemy):...
    def die(self):...
    def show_inventory(self, inventory):...
    def unequip(self, item):...
    def equip(self, item):...
    def consume(self, item):...
```

Рисунок 5 – Класс `Player`

Очки здоровья, Урон, Защита и Ловкость находятся в словаре. Уровень, опыт, очки развития и локация – в отдельных переменных. Инвентарь – список.

### 1.2 Локации, враги и предметы

Локации, встречаемые на них враги и находимые предметы находятся в словаре `wiki_locations`. Он представлен на рисунке 6.

```
wiki_location = {
    'Лес': [
        ["Бабочка", "Гоблин", "Гоблин-шаман", "Орк", "Голубая слизь", "Розовая слизь", "Красная слизь", "Олень-перевёртыш", "Дриада"],
        ["Рог оленя-перевёртыша", "Серьга гоблина", "Накидка из шкуры оленя-перевёртыша",
        "Диадема лесной дриады", "Ножи разбойника", "Клинок. Просто клинок.", "Красное яблоко", "Пятнистый гриб"],
        ["", ""],
        ["- сМотРи, ТаМ ЧТо-То ВнутРи дУпЛа?", "- В ЭтоМ кУсте КажЕтся ЧТо-то есть..."]],
    ...
}
```

Рисунок 6 – Локации

Враги генерируются по мере необходимости за счет класса `Creature`, а все необходимые входные данные для разных типов (Очки здоровья, Урон, Ловкость, Защита, оставляемые при смерти лут и опыт) содержатся в словаре `wiki_monsters`, который представлен на рисунке 7. Их уровень зависит от уровня самого Игрока.

```
wiki_monsters = {
    "Бабочка": [(1,2), (1,1), (13,18), (0,0), (0,5), 17],
    "Гоблин": [(3,6), (3,6), (2,5), (2,5), (2,4), 15,
               ['Серьга гоблина', 'Ягодный сок', "Кость"]],
    "Гоблин-шаман": [(3,6), (4,10), (2,4), (2,5), (3,5), 23,
                     ['Плащ шамана-гоблина', 'Книга заклинаний', 'Ягодный сок']],
    "Орк": [(10,19), (6,8), (2,4), (10,12), (13,16), 35,
            ['Красное яблоко', 'Пятнистый гриб', ]],
    "Стеснительный орк": [(25,36), (7,10), (2,4), (12,15), (32,50), 54,
                           ['Маска стеснительного орка', 'Красное яблоко', 'Дубинка стеснительного орка']],
}
```

Рисунок 7 – wiki\_monsters

Характеристики предметов содержатся в двух словарях – wiki и wiki\_consume. Первый включает предметы, которые можно надеть, второй – все предметы, их описание и получаемые улучшения при их поглощении.

### 1.3 Цикл игры

Игра работает за счёт цикла, который выполняется, пока Здоровье Игрока не равно нулю. Он запускает функцию menu(), которая содержит в себе ряд других. Сначала происходит случайная генерация мест, куда может отправиться протагонист через gen\_location(), затем непосредственно выбор со стороны Игрока в location(). В зависимости от этого выбора раунд игры может завершиться или продолжиться. При продолжении функция gen\_mon() случайным образом заполняет список cur\_mon соответствующими этой локации монстрами в количестве от одного до трёх. Потом они перечисляются через mon\_meet() и начинается битва battle(), в которой поочерёдно воспроизводится attack() и mon\_attack(). После смерти монстры удаляются из cur\_mon, и когда он оказывается пуст, повторение battle() прекращается, и, если Игрок не проиграл, начинается новый цикл.

## Заключение

В результате выполнения проекта была разработана и реализована текстовая ролевая игра (RPG) для консоли на языке Python. Основная цель работы — создание интегрированной программной системы, моделирующей ключевые игровые процессы, — достигнута.

Были успешно реализованы все запланированные функциональные модули:

- 1) система создания персонажа, включающая генерацию характеристик
- 2) пошаговая боевая система с расчётом урона и вариативностью действий игрока
- 3) механика получение опыта, повышение уровня и характеристик персонажа
- 4) система инвентаря для управления предметами, влияющими на игровой процесс
- 5) модуль локаций, обеспечивающий базовое нелинейное перемещение

Все модули объединены в единый цикл, что обеспечивает устойчивую и предсказуемую работу приложения. В процессе разработки были применены фундаментальные парадигмы программирования: объектно-ориентированный подход для моделирования сущностей и процедурный — для организации основного игрового цикла и логики. Выбор структур данных (списки, словари) был обусловлен требованиями к частоте обращения и модификации информации.

Таким образом, проект представляет собой законченное консольное приложение, которое демонстрирует практическое применение стандартных средств языка Python для построения структурированной пошаговой игровой системы. Код проекта может служить основой для дальнейшего расширения, такого как увеличение разнообразия контента, усложнение игровых механик или адаптация архитектуры для добавления графического интерфейса.